BRANCH PREDICTOR USING BRANCH PREDICTION ACCURACY HISTORY

This application relies for priority upon Korean Patent Application No. 1999-45786, filed on October 21, 1999, the contents of which are herein incorporated by reference in their entirety.

Field of the Invention

The present invention relates to the field of computer system, and more particularly to a branch predictor using branch prediction accuracy history, efficient processing techniques for instruction streams which include conditional program flow instructions, such as branch instructions.

Background of the Invention

Many microprocessors employ a technique known as hardware pipelining to increase instruction throughput by processing several instructions through different phases of execution concurrently. To maximize instruction execution efficiency, it is desirable to keep the instruction execution pipeline full (with an instruction being processed in each pipeline stage) as often as possible such that the pipeline produces useful output every clock cycle. However, whenever there has been a transfer of program flow control to another section of software code and instructions have been speculatively fetched and processed and it is determined that these instructions should not have been executed, the output from the pipeline is not useful.

Exceptions and program flow control instructions such as branch instructions, provide examples of how the program flow control can be changed. Branch instructions, which may be conditional or unconditional and may transfer program flow control to a preceding or subsequent code section, are used for frequency encountered situations where a change in program flow control is desired.

A conditional branch instruction determines instruction flow based on the resolution of a specified condition. “If A>B then branch to instruction X” is an example of a conditional branch instruction. In this case, if A>B, program flow control branches to a code section beginning with instruction X, also referred to as the target code section. If A is not greater than B, the instructions sequentially following the branch instruction in the program flow, referred to as the sequential code section, are to be executed. For executing such conditional branch instruction, it is required to check a condition of the branch instruction for determining the next instruction. Thus, performance of a microprocessor including a central processing unit (CPU) may be deteriorated in pipeline procedures of the microprocessor requiring fast instruction fetch.

For solving the aforementioned problem, many microprocessors adopt a branch predictor (or a branch prediction logic), which operates to predict the outcome of a branch instruction before identifying a condition check of the branch instruction, based on a predetermined branch prediction approach. Thus, instructions are then speculatively fetched from either the target code section or the sequential code section based on the prediction indicated by the branch predictor. Therefore, a pipeline stall can be prevented. However, when a branch prediction is missed, many instructions from the incorrect code section may be in various stages of processing in the instruction execution pipeline. On encountering such a misprediction, instructions following the mispredicted conditional branch instruction in the pipeline (or multiple pipelines) are flushed, and instructions from the other, correct code section are fetched. Flushing the pipeline creates bubbles or gaps in the pipeline. Several clock cycles may be required before the next useful instruction completes execution, and before the instruction execution pipeline produces useful output. Such incorrect guess causes the pipeline to stall until it is refilled with valid instructions, this delay is called the mispredicted branch penalty.

To reduce above described misprediction ratio, various kinds of branch predictors are used. Among the branch predictors, a two-level branch predictor is likely to become more common. A P6 processor of Intel Corporation is the first to use a two-level branch algorithm to improve accuracy. This algorithm, first published by Tse-Yu Yeh and Yale Patt, has the potential to push accuracy well beyond the 90% level achieved by the best processors today.

Fig. 1 is a schematic diagram for illustrating a structure of a conventional two-level branch predictor. For example, the branch predictor is illustrated in Fig. 2 of "*New Algorithm Improves Branch Prediction*" by Linley Gwennap, March 27, 1995, MOCROPROCESSOR, pp. 17-21.

Referring to **Fig. 1**, the two-level branch predictor is composed of a branch history register (BHR) **10** and a pattern history table (PHT) **20**. The branch history register **10** is used for recording the actions of the most recent k conditional branches. For example, a "1" stored in the branch history register **10** may denote a branch "taken", and a "0" stored in the branch history register **10** may denote a branch "not taken", respectively. The performed k conditional branches are called a pattern.

The pattern history table **20** is used for recording a pattern history bit **Sc**, which is used for predicting a conditional branch of a branch instruction to be performed in response to each pattern. For example, the two-level branch predictor predicts a conditional branch **I(Sc)** in response to an entry of "10" stored in the pattern history table **20**. The entry corresponds with a pattern "111010" stored in the branch history register **10**. According to the predicted conditional branch **I(Sc)**, the next instruction to the branch instruction is fetched. Referring to the paper of the Linley Gwennap, a predicted conditional branch **I(Sc)** is determined by a most significant bit (MSB) of a pattern history bit **Sc** stored in the pattern history table **20**.

For example, on the assumption that a real conditional branch of the branch instruction is **Rc**, if a predicted conditional branch **I(Sc)** is different from the real conditional branch **Rc**, this case is called a prediction miss. In this case, execution of instructions following the mispredicted conditional branch **I(Sc)** are withdrew.

According to the real conditional branch **Rc**, both data of the branch history register **10** and the pattern history bit **Sc** stored in the pattern history table **20** are changed. This process is described as follows.

When a least significant bit (LSB) corresponding to the real conditional branch **Rc** of the branch instruction is stored to the branch history register **10**, the rest bits are shifted to left. At this time, the pattern history bit **Sc** stored in the pattern history table **20** is updated in response to the real conditional branch **Rc**. For example, if the real conditional branch **Rc** is "1" denoting "predict taken", the pattern history bit **Sc** is increased of "1", and if the real conditional branch **Rc** is "0" denoting "predict not taken", the pattern history bit **Sc** is decreased of "1". The pattern history bit **Sc** can be composed of an up/down saturating counter as shown in "*A Study of Branch Prediction Strategies*", by J. Smith, May 1981, pp. 135-148. The saturating counter maintains a minimal value of a pattern history bit **Sc** when the pattern history bit **Sc** is the minimal value, although the real conditional branch **Rc** is "0" denoting "not taken". In addition, the saturating counter maintains a maximum value of a pattern history bit **Sc** when the pattern history bit **Sc** is the maximum value, although the real conditional branch **Rc** is "1" denoting "taken".

Although branch prediction accuracy may be improved or turned by using different branch prediction algorithms, mispredictions still occur. By the time a misprediction is identified, many instructions from the incorrect code section may be in various stages of processing in the instruction execution pipeline.

An example to solve the forgoing performance penalty relavant to mispredicting is disclosed in U.S. Pat. No. 5,860,017 to Sharangpani et al., issued on Jan. 12, 1999, entitled "*PROCESSOR AND METHOD FOR SPECULATIVELY EXECUTING INSTRUCTIONS FROM MULTIPLE INSTRUCTION STREAMS INDICATED BY A BRANCH INSTRUCTION*", which identifies branch instructions, which in relationship to other conditional branch instructions, have a relatively high likelihood of being mispredicted. In this case, once a condition in a branch instruction is identified as being unlikely to be predicted accuracy, the processor fetches and decodes instructions from both target and sequential instruction streams indicated by the conditional branch instruction. However, the method proposed by Sharangpani et al. may occur performance deterioration by a resource conflict and may lead to high hardware cost, since the processor fetches both target and sequential instruction streams. Therefore, a novel branch predictor is required capable of efficient processing of branch instructions by reducing prediction miss with simple circuit configuration and low hardware cost.

Summary of the Invention

It is therefore an object of present invention to provide a branch predictor capable of efficiently processing branch instructions by reducing prediction misses with simple circuit configuration and low hardware cost.

In order to attain the above objects, according to an aspect of the present invention, there is provided a branch predictor comprising: branch prediction means for predicting a conditional branch of a branch instruction; a comparator for generating a comparison signal by comparing the predicted conditional branch from the branch prediction means with a real conditional branch of the branch instruction; an accuracy history table for storing am accuracy history of the predicted conditional branch; a first state transition logic for generating an accuracy history bit to be stored to the accuracy history table in response to the comparison signal; and a multiplexer for outputting either the conditional branch or an inverted conditional branch as a final branch prediction outcome, in response to a predicted accuracy history signal based on the accuracy history bit.

Brief Description of the Drawings

A more complete appreciation of the present invention, and many of the attendant advantages thereof, will become readily apparent as the same becomes better understood by reference to the following detailed description when considered in conjunction with the accompanying drawings in which like reference symbols indicate the same or similar components, wherein:

Fig. 1 is a schematic diagram for illustrating a structure of a conventional two-level branch predictor; and

Fig. 2 is a schematic diagram for illustrating a structure of a two-level branch predictor according to the present invention.

Description of the Preferred Embodiment

It should be understood that the description of this preferred embodiment is merely illustrative and that it should not be taken in a limiting sense. In the following detailed description, several specific details are set forth in order to provide a thorough understanding of the present invention. It will be obvious, however, to one skilled in the art that the present invention may be practiced without these specific details.

A branch predictor outputs either a predicted conditional branch or an inverted predicted conditional branch as a final branch prediction outcome, in response to a predicted accuracy history signal based on an accuracy history bit. According to the accuracy history bit, it is determined whether the branch prediction outcome of the branch predictor is correct or not. If the predicted conditional branch is correct, the branch predictor outputs the predicted conditional branch, and if the predicted conditional branch is not correct, the branch predictor outputs the inverted predicted conditional branch, in response to the predicted accuracy history signal.

Fig. 2 is a schematic diagram for illustrating a structure of a two-level branch predictor according to the present invention. Referring to **Fig. 2**, the two-level branch predictor comprises a branch history register **15** for recording actions of the most recent k conditional branches, a pattern history table **25** for recording a pattern history bit **Sc** used for generating a predicted conditional branch **I(Sc)**, and an accuracy history table **60** for recording accuracy history of the predicted conditional branch **I(Sc)**. The accuracy history table **60** is composed of a memory array.

To the pattern history table **25**, a first state transition logic **30** generating a pattern history bit **Sc** to be stored to the pattern history table **25** in response to a real conditional branch **Rc**, is coupled. In addition, to the accuracy history table **60**, a second state transition logic **50** generating an accuracy history bit **Ac** to be stored to the accuracy history table **60**, is coupled.

Further, the branch predictor according to the present invention comprises a comparator **40** generating a comparison signal by comparing the predicted conditional branch **I(Sc)** generated by the pattern history bit **Sc** with the real conditional branch **Rc** of the branch instruction. The comparison signal is inputted to the second state transition logic **50** to generate the accuracy history bit **Ac**. In addition, the branch predictor comprises a multiplexer **70** selecting either a predicted conditional branch **I(Sc)** or an inverted predicted conditional branch as a final branch prediction outcome. A predicted accuracy history signal **I(Ac)** based on the accuracy history bit **Ac** is used as a selection signal for the multiplexer **70**. Operation of the branch predictor is described as follows.

First of all, a predicted conditional branch **I(Sc)** is generated in response to a pattern history bit **Sc** corresponding to a pattern stored in the branch history register **15**. The predicted conditional branch **I(Sc)** is inputted to the comparator **40** to be compared with a real conditional branch **Rc**.

The real conditional branch **Rc** has a "1" or "0" value according to "predict taken" or "predict not taken", and the value stored in the branch history register **15** is updated in response to the value of the real conditional branch **Rc**. According to the updated value of the branch history register **15**, the pattern history bit **Sc** is updated. The first state transition logic **30** updates the pattern history bit **Sc**. The first state transition logic **30** is composed of an up/down saturating counter. In the first state transition logic **30**, the value of the pattern history bit **Sc** is increases by 1 when the real conditional branch **Rc** is "1" (i.e., taken), and the value of the pattern history bit **Sc** is decreased by 1 when the real conditional branch **Rc** is "0" (i.e., not taken).

The predicted conditional branch **I(Sc)** has a value of "1" or "0" in response to a most significant bit (MSB) of the pattern history bit **Sc**. The comparator **40** outputs "1" or "0" as a comparison signal to the second state transition logic **50** by comparing the real conditional branch **Rc** and the predicted conditional branch **I(Sc)**. For example, if the predicted conditional branch **I(Sc)** is the same as the real conditional branch **Rc**, the comparator **40** outputs "1", and if the predicted conditional branch **I(Sc)** is different from the real conditional branch **Rc**, the comparator **40** outputs "0".

The second state transition logic **50** receiving the comparison signal determines an accuracy history bit **Ac** to be stored to the accuracy history table **60** in response to the comparison signal. The second state transition logic **50** is composed of an up/down saturating counter increasing the value of the accuracy history bit **Ac** by 1 when the predicted conditional branch **I(Sc)** is the same as the real conditional branch **Rc**, and decreasing the value of the accuracy history bit **Ac** by 1 when the predicted conditional branch **I(Sc)** is different from the real conditional branch **Rc**. The accuracy history bit **Ac** can be used after learning a branch accuracy of corresponding pattern by monitoring the pattern.

According to the above described method, the accuracy history bit **Ac** is determined and stored to the accuracy history table **60**. According to the accuracy history bit **Ac**, it can be determined whether a prediction result of the branch predictor is correct or not.

For example, if a pattern history bit **Sc** is "011" corresponding to a pattern "11...10" stored in the branch history register **15**, a predicted accuracy history signal **I(Ac)** is generated by an MSB of the accuracy history bit **Ac**. The predicted accuracy history signal **I(Ac)** is used for determining whether the predicted conditional branch **I(Sc)** is correct or not. For example, if it is considered as the predicted conditional branch **I(Sc)** is correct, the predicted accuracy history signal **I(Ac)** having a value of "1" is outputted to the multiplexer **70**. Thus, the predicted conditional branch **I(Sc)** is outputted from the multiplexer **70** as a final prediction result. In addition, if it is considered as the predicted conditional branch **I(Sc)** is not correct, the predicted accuracy history signal **I(Ac)** having a value of "0" is outputted to the multiplexer **70**. Thus, the inverted predicted conditional branch is outputted from the multiplexer **70** as a final prediction result. As described above, the predicted accuracy history signal **I(Ac)** is used as a selection signal of the multiplexer **70** selecting either the predicted conditional branch **I(Sc)** or an inverted predicted conditional branch as a final prediction outcome of the branch predictor.

As described above, the branch predictor according to the present invention outputs either a predicted conditional branch or an inverted predicted conditional branch as a final branch prediction outcome, in response to a predicted accuracy history signal based on an accuracy history bit, so that the two-level branch predictor can reduce the misprediction and a microprocessor can process branch instructions more efficiently. In this case, the branch prediction according to the present invention merely appends the accuracy history table **60** and multiplexer **70** to the conventional branch predictor. Thus, the branch prediction according to the present invention can reduce the misprediction with simple circuitry structure and low hardware cost.

While the invention has been described in terms of an exemplary embodiment, it is contemplated that it may be practiced as outlined above with modifications within the spirit and scope of the appended claims.